Applied Microeconomics: Problems in Estimation, Forecasting, and Decision-Making

Instructor's Manual

Richard L. Schmalensee

July, 1970 471-70

Applied Microeconomics: Problems in Estimation, Forecasting, and Decision-Making

Instructor's Manual

Richard L. Schmalensee

July, 1970 471-70

CONTENTS

CHAPTER I

Introduction

Applied Microeconomics presents a sequence of exercises designed to supplement intermediate level courses in microeconomic theory or managerial economics. A working knowledge of elementary calculus and the availability of a standard computer program for multiple regression are assumed. The tools and concepts needed for the exercises are discussed in the Student's Manual; the computer programs to be employed by the instructor, except the regression routine, are presented here.

The problems presented here are, of course, simpler than most encountered in the real world. They are not intended to convey all of the difficulties encountered in actual empirical work. Rather, the focus is on learning to use powerful tools and concepts in simple but nontrivial situations. These same approaches can be applied to more complex problems, though not as easily as they can be used here. The hope is that working on fairly simple problems will make it easier for the student when he is faced with a more complicated situation of the same basic type.

The first two exercises involve using multiple linear regression to analyze computer-generated data. Chapter III describes a demand estimation exercise, and Chapter IV outlines a cost estimation problem. A computer program called DBEAST generates the demand data, while CBEAST generates cost data. The demand and cost curves and the relations between

the curves estimated in these two problems and the other exercises are discussed in Chapter II.

The next exercise consists of using the estimated demand and cost schedules to forecast short-run and long-run industry equilibrium under monopolistic and competitive conditions. This exercise is discussed in Chapter V. Industry equilibria are computed (using the true demand and cost schedules) by a program called IGAME.

The IGAME program is also used to initialize the last exercise, an oligopoly price determination game. The maket structures that may be simulated are discussed in Chapter V. Students' pricing decisions will be based on their demand and cost curves and on the results of the forecasting exercise. The game is run using the RGAME program, dicussed in Chapter VI. Finally, Chapter VII outlines the program SGAME, used to summarize the results of several periods of play of the game.

A wide variety of cost and demand schedules can be employed, and IGAME permits the instructor to simulate a number of different maket structures. I think there are moderate research possibilities here. It is possible to examine experimentally the impact of various dimensions of maket structure on market performance.

The programs described herein were written in the summer of 1969 by Richard Butler, who also helped me produce this manual. They were slightly modified during the 1969-70 school year by Walt Maling. I would like to thank them both for a job well done. All programs are written in FORTRAN IV; all have been tested on the IBM 360/65 at M.I.T. The source programs, extensively commented, are punched on just under one and a third boxes of cards.

The Sloan School of Management at M.I.T. and the Edwin Land Foundation provided considerable financial support. I am indebted to Edwin Kuh and Paul MacAvoy in countless ways. Finally, I must thank all the students who have suffered through earlier versions of these exercises. They may have learned something; I certainly did.

CHAPTER II

Basic Structure

Demand Curves

All demand curves in these exercises are linear. Data are generated by DBEAST according to the following function:

$$(2.1) \qquad Q = A + BY + DP + RMU; \quad A > 0, \quad D < 0.$$

The variable Q is units sold per firm, Y is income, P is industry price, and RMU is a normally-distributed error term with zero mean and user-supplied standard deviation. The constants A, B, and D are user-supplied. The demand data refers to the sales of a typical firm in a situation where all firms in the industry charge the same price.

In the forecasting exercise, students are told the number of firms originally present in the industry, N, and they are given a value of income that will hold for the rest of the exercises. Call this value YMID. (The income level is given by DBEAST; the number of firms is arbitrary.) Then the industry demand curve used by IGAME to compute the answers to the forecasting exercise is

$$(2.2) \qquad QI = NA' + NDP; \quad \text{where } A' = A + BYMID > 0$$

and where QI is industry sales. Once N and YMID are given, the students should be able to convert their estimated firm demand schedules, estimates of (2.1), into industry demand curves, estimates of (2.2).

In the oligopoly game, each firm will generally be facing a demand schedule of the following form:

$$(2.3) \qquad Q = A' + E\,P + F\,\overline{PO} + RMU; \; E + F = D,\; E < 0,\; F > 0.$$

Here P is the firm's price, and $\overline{PO}$ is the unweighted arithmetic average of the prices charged by the other firms in the industry. Subject to the constraint on their sum, E and F are specified by the instructor. The larger these constants are, the more sensitive market shares are to price differentials. Notice that when $P = \overline{PO}$, equation (2.3) becomes equation (2.2) divided by N. Thus when all firms charge the same price, the demand schedule facing each one is as estimated in the estimation problem. Notice also that equation (2.3) may be re-written so that each firm's sales is a linear function of its price and of the unweighted average price for the industry as a whole.

During the game, entry into or exit from industries may occur. This changes the demand curve facing each firm from (2.3) to

$$(2.4) \qquad Q = f\,A' + f\,E\,P + f\,F\,\overline{PO} + f\,RMU,$$

where f = (original number of firms)/(current number of firms). Thus if the industry average price is unchanged, entry or exit will leave (average) total sales unchanged. The constant f is normally equal to one at the start of the game, but industries may be set up with high or low demand schedules by varying f before the game starts. This is discussed in Chapter V, below. Notice that the error term, RMU, is also multiplied by f. A smaller absolute demand curve implies less absolute variation for each firm. All firms operating in the same industry face the same demand curve, except for

the error term.

The demand estimation exercise does not provide all the information needed to play the game. Students' estimates tell them how industry and firm demand will vary when all firms charge the same price. That is, they have estimated D in (2.1) and (2.2). But experience in the game is necessary to acquire information on the cross-price elasticity within the industries, to estimate E and F in (2.3).

Cost Functions

Data are generated by CBEAST for the cost function for a typical firm. All firms own one plant, and all plants in a given industry have identical cost functions. All plants in all industries have a capacity of 100,000 units. At zero output, all cost functions have a marginal cost equal to $.50 during the game. When firms produce capacity output, marginal cost may have any value. The user supplies the capacity marginal cost that is to prevail during the game, and the program computes the constants a and b from the assumed form of the marginal cost function:

$$(2.5) \qquad MC = a + bQ^2,$$

where Q is units of output of the plant, and MC is marginal cost.

If we denote fixed cost by FC, the expression for total cost that prevails during the game is

$$(2.6) \qquad TC = FC + a\,Q + (b/3)\,Q^3 + RTC,$$

where RTC is normally distributed with zero mean and user-supplied standard deviation. The fixed cost, FC, is normally $50,000, but the

user may request (in CBEAST and IGAME) that it be adjusted so that the short-run competitive equilibrium involves zero profit. (It will then be identical to the long-run competitive equilibrium.)

In order to make the problem of estimating a and b more interesting, two complications have been introduced into CBEAST. First, observations on quantity are not given explicitly. Instead, the program provides price and total revenue. The relation between price and quantity to be used in these computations is the demand curve that is to prevail during the play of the game; its parameters are user supplied.

The second complication is much less a red herring than the substitution of price and revenue data for quantity information. We take as exogenous to the system an index of the price of the variable inputs, which are assumed to be used in fixed proportions. The user supplies the maximum and minimum values of this factor price index, FPI, and it is assumed that FPI will be fixed at the midpoint of the specified range, FPMID, during the game. The expression for total cost empolyed by CBEAST makes variable cost proportional to FPI:

$$(2.7) \qquad TC = FC + \left[\frac{a\ 100}{FPMID}\right]\left[\frac{Q\ FPI}{100}\right] + \left[\frac{b\ 10^{12}}{3\ FPMID}\right]\left[\frac{Q^3\ FPI}{10^{12}}\right] + RTC$$

$$= FC + a'\ W1 + b'\ W2 + RTC$$

The constant 100 appears in a' because that value seems an obvious choice for FPMID. The constant b is normally quite small; it is multiplied by 10^{12} to make b' comparable with a'. This is done primarily to guard against roundoff error in computing regression results.

The Exercises

In the simplest case, DBEAST is used to produce one set of demand data and CBEAST is employed to create one set of cost data. Students then estimate firm demad and cost curves. They are then given YMID, FPMID, and the number of firms in the industry, and they estimate industry equilibria. IGAME is used by the instructor to calculate the true equilibria. Then, in the game, all firms generally begin with the same basic demand and cost situation. It is possible, of course, to vary the sensitivity of demand to intra-industry price differentials; all industries need not have the same E and F, even if they do have the same D.

Each firm is identified in the game by a four-digit number. The first two digits refer to the industry the firm began in, and the second two are the firm's number within the industry. If there are to be three industries, they must be numbered 01, 02, and 03. If industry 03 begins with four firms, they must be given firm numbers 0301, 0302, 0303, and 0304. Experience suggests that no more than four students should be placed in each firm.

If students are assigned firm numbers at the start of the exercise, more complicated variants of the above sequence can be envisioned. Suppose, for instance, that eight industries are set up. Then two sets of demand and cost data could be generated, one set for industries 1 - 4 and the other set for industries 5 - 8. Students might be required to estimate only the structure for their industry, though it might be useful to require them to examine both sets of data. In any case, if firms are to be allowed to switch industries (see Chapter V), students should be

permitted to examine both sets of data in order to make intelligent entry or exit decisions.

In moving from the estimation exercises to the forecasting problem, it must be emphasized that the students have estimated the cost and demand curves of a typical plant in a situation in which all firms charged the same price. All that is necessary is to understand the distinction between firm and industry demand. In the game, the same story applies, except that the number of firms will in general be different from the number assumed present in the forecasting problem. Industry demand depends on the unweighted average of firm prices. If all firms charge the same price, the estimated demand curve applies to each. A change in the number of firms changes the demand curve facing each firm, but industry demand is unchanged. Thus the industry demand curve is determined by the firm schedule estimated, the original number of firms, and the parameter f in equation (2.4).

A numerical example (with numbers that bear little relation to those actually used) may help clarify this point. Suppose the equation used by DBEAST is

$$Q = 100 + 2Y - 3P + RMU.$$

If YMID = 10 and the forecasting exercise is to be done with a twenty firm industry,the industry demand equation is

$$QI = 2400 - 60 P.$$

If a student finds himself in a three-firm industry in the game, the

industry demand curve he faces is

$$QI = 360 - 9\overline{P}.$$

This will not be changed by entry or exit of firms. If there are currently n firms in this industry, the demand curve facing each one will be

$$Q = 360/n + Z\,\overline{PO}/n - (9 + Z)\,P/n,$$

where the positive constant Z is determined by the instructor at the start of the game. Note that if $P = \overline{PO}$ and $n = 3$, this reduces to the DBEAST equation with Y = Y'MID.

CHAPTER III

Demand Estimation

Introduction

This chapter describes the demand estimation exercise. We first examine the structure and use of the DBEAST program, which produces the data to be analyzed. The chapter concludes with a brief discussion of the design and grading of the exercise.

The basic form of the demand function used by DBEAST is given by equation (2.1), which we re-write here for convenience:

$$(3.1) \qquad Q = A + B\,Y + D\,P + RMU.$$

The variable Q is units sold per firm, Y is income, P is industry price, and RMU ia a normally distributed error term with zero mean and user-supplied standard deviation. The coefficients A and D are calculated internally by the program; B is user-supplied. (The price coefficient, D, is negative.) This demand curve is designed to be used in the game with Y fixed at its mean value, YMID. Given YMID and the price a monopolist would charge with marginal cost of $.50 and Y = YMID, PM, DBEAST computes A and D.

The user supplies a maximum and minimum allowable price, and the program computes a similar range for income, in a manner to be discussed below. Each quantity observation is generated by selecting values for income and price and adding an error term. The first-order autocorrelation

coefficient (smoothness) of the income and price series are specified by the user.

Program Input

An input deck to DBEAST is made up as follows:

1. A Title Card with any comment punched in columns 1 - 80. This comment will appear at the top of each page of output.
2. One or more coefficient decks, each of which consists of a Coefficient Card followed by one or more Trial Cards.
3. A Kill Card with zeros punched in columns 1 and 2.

The format of the Coefficient Cards is shown in Table III.1. If KP is one, all data generated by the trials made with this set of coefficients will be punched, one observation to a card. If KP is two, each variable will be punched as a separate deck. If KS is one, the user must supply a starting point or "seed" for the random number generation routine on each trial card that follows this coefficient card. Usually, one will want to run a number of trials without punching data, to get a feel for the regression results different parameters produce. Once a good set of parameters has been located by this procedure, a deck will be punched and turned over to the students. The KS feature permits the user to duplicate the random sequence used in the first set of runs, since each trial's output contains the seed that was used to start it. A seed may be any odd integer, preferably containing four or more digits. If KS is zero, a seed is supplied by the program at the beggining of each run and is used to generate a different seed for each trial.

Monopoly price will typically be in the range \$2.00 - \$9.00; this is the range for which the game was designed. The income coefficient, BH, can have any value, though it should probably be greater than one. Income, Y, is best thought of as aggregate disposable income in billions of dollars. Thus YMID should be well over 100.

The format for Trial Cards is shown in Table III.2. The income range is calculated so that less than 16% of all observations will produce a Q below QMIN. If such a Q is calculated, it is dropped from the sample, and the program goes on to calculate another observation. In the unlikely event that more than NOB Q's below QMIN are encountered in the course of generating NOB observations, the trial is terminated and an error message printed.

It can be the case that there is no income range compatible with the input parameters. This occurs when the minimum income (the income for which fewer than 16% of Q's will be below QMIN) is greater than or equal to YMID. If this occurs, the program will print an error message indicating the highest permissable PMAX given the other parameters. Table III.3 provides guidance on the choice of parameters. The values of PMAX used must be less than those shown in the Table for given PM and QMIN. Note that QMIN = 0 allows the most freedom in picking PMAX.

Data Generation

Observations on price, quantity, and income are generated according to the following formulae, where P(n) is the value of P in the n^{th} observation:

$$P(n) = (1 - ACCP)(PMIN + PRANGE \cdot PRAN) + ACCP \cdot P(n - 1)$$

$$Y(n) = (1 - ACCY)(YMIN + YRANGE \cdot YRAN) + ACCY \cdot Y(n - 1)$$

$$Q(n) = A + BY(n) + DP(n) + RMU$$

The quantities PRANGE and YRANGE are the ranges of price and income, equal to the difference between the maximum and minimum permissable values. PRAN and YRAN are random numbers uniformly distributed between zero and one. It should be clear that larger auto-correlation coefficients (ACCP and ACCY) will produce smoother price and income series. RMU is normally distributed with mean zero and standard deviation SD = SDI x 1000. The constant B is equal to BH x 100, where BH is supplied by the user. The parameters A and D are calculated according to

$$A = \frac{100,000\ (4\ PM - 1)}{(4PM - 3)} - B\ YMID$$

$$D = \frac{200,000}{4PM - 3}$$

Minimum income, YMIN, is calculated according to

$$YMIN = (QMIN - A - D\ PMAX + SD)/B.$$

Table III was constructed for SD = 0, its smallest possible value.

<u>Program Output</u>

All observations generated are printed. In addition, all important input parameters and calculated parameters are shown. These numbers include monopoly price, income coefficient, ranges and autocorrelation coefficients for price and income, minimum quantity, the

standard deviation of the error term. In addition, the seed for the random sequence used is presented, in case the user wishes to reproduce the exact observations generated.

The next block of printed outputs gives the true values of A, B, and D as well as the values that would be produced by estimating equation (3.1) from the generated data. The usual statistics provided by a regression package are also presented.

Finally, the means of quantity, price and income are shown. All of this information is given to enable the user to decide quickly whether the set of data just generated is suitable for student use and to remind him of the inputs that went into that data. It should be a fairly simple matter to change the inputs to produce useable numbers.

One reminder: The R-Squared gives no important information about the usefulness of the data to students. What matters is the quality of the coefficient estimates. If the estimated coefficients are significant and are near their true values, the students will be able to use the estimated demand function successfully in the game.

If a deck is requested (if KP = 1 or 2 on the trial card), the generated observations will be punched as specified by KP. A title card stating "THIS CARD INDICATES THE BEGINNING OF A DATA SET" will be punched before the observations for each trial.

Subroutines Employed

In calculating regression output, DBEAST employs two generalized matrix manipulation subroutines written by Robert Hall. MULT performs

matrix multiplication, and GMINV inverts square matrices.

RANDU is an IBM Scientific Subroutine Package program which generates quasi-random numbers distributed uniformly between zero and one. Input to the routine is an odd integer; output is the desired random number plus a second odd integer which is used as an input the next time the subroutine is called. This feature makes it possible to reproduce the sequence of random numbers generated. This subroutine is specific to the IBM System/360; a new routine must be written if DBEAST is to be used on any other computer.

RANGRD obtains uniformly distributed random numbers from RANDU, places two of them in common where the main program can find and use them as PRAN and YRAN, and uses two others to generate a normally-distributed random number to serve as RMU. This routine is not specific to System/360.

The Exercise

Regression programs tend to be rather complicated, and many of the better ones work well only on the machine for which they were originally designed. Hence none is presented in this manual. The mechanics of using the data generated by DBEAST and by CBEAST will, however, depend on the regression routine employed. In a time-sharing environment, the data can simply be stored where all students can access it. Some batch processing systems will permit the use of only one data deck per run, with students' control cards following. On other systems, it will be necessary for each student to have his own data deck. Careful thought should be given this

problem well before the students are to make their runs.

The program generates observations on P, Y, and Q. These need not be the series explicitly given the students, though you may wish to construct revenue from P and Q and then provide revenue and either P or Q. Also, it is often interesting to insert an unrelated series, say the P series from another run of DBEAST, and label it the price of a related good. Normally, one would like this unrelated variable to show up with an insignificant coefficient in the regression, so that students should drop it from consideration. A run should be made with any candidate fourth series before giving data to students to ascertain whether its coefficient is insignificant. All data given to students should be labeled carefully, with units indicated.

Short papers should be required of individual students, in which they discuss the results of their estimations. (Each student should estimate at least three equations.) Grading should be based on the understanding of both statistical and economic tools. The key statistics from the computer should be discussed, and the preferred demand function should be defended statistically. But it is equally important that the demand functions discussed make economic sense. Occam's razor should be emphasized. While originality should be encouraged, complex constructs with no economic meaning should be criticized. Equations should be simple and sensible. Finally, it seems both easy and important to require students to derive and discuss the elasticities of demand. (Elasticities should be evaluated at the point of sample means.)

Students tend to try independent variables one at a time,

retaining ones that "work". It should be pointed out early on that this procedure can give seriously misleading results. Similarly, estimation of identities or near-identities should be vigorously discouraged.

Table III.1

Format of DBEAST Coefficient Cards

Columns	Variable	Description
1 - 2	NT	Number of trials to be made with this set of coefficients (right-justified, no decimal point).
3	KP	Punch a 1 if data generated with these coefficients is to be punched; punch a 0 if no punched output is requested from these trials.
4	KS	Punch a 1 if the following trial cards contain seeds for the random number routine. If 0 is punched, the program will supply starting terms in the random sequences.
5 - 14	PM	The monopoly price when marginal cost is constant and equal to $.50 and Y = YMID (punch a decimal point).
15 - 24	BH	The income coefficient in the demand function, in hundreds (punch a decimal point).

Table III.2

Format of DBEAST Trial Cards

Columns	Variable	Description
1 - 3	NOB	The number of observations (less than or equal to 200) to be generated in this trial. (right-justified, no decimal point)
4 - 11	PMIN	The lower end of the price range (punch a decimal point).
12 - 19	PMAX	The upper end of the price range (punch a decimal point).
20 - 27	ACCP	The autocorrelation coefficient for price (punch a decimal point). Must be between zero and one.
28 - 35	QMIN	The lower bound on quantity (punch a decimal point).
36 - 43	YMID	The midpoint of the income range (punch a decimal point).
44 - 51	ACCY	The autocorrelation coefficient for income (punch a decimal point). Must be between zero and one.
52 - 59	SDI	Standard deviation of the error term, RMU, in thousands (punch a decimal point).
60 - 71	KU	Seed for random number routine. KU will not be read unless KS = 1 on the coefficient card. KU must be an odd integer; punch it right-justified without a decimal point.

Table III.3

Maximum Values of PMAX in DBEAST

	PM			
QMIN	2.00	3.00	4.00	5.00
0	3.50	5.50	7.50	9.50
10,000	3.25	5.05	6.85	8.65
20,000	3.00	4.60	6.20	7.80
30,000	2.75	4.15	5.55	6.95
40,000	2.50	3.70	4.90	6.10
50,000	2.25	3.25	4.25	5.25
60,000	2.00	2.80	3.60	4.40
70,000	1.75	2.35	2.95	3.55
80,000	1.50	1.90	2.30	2.70
90,000	1.25	1.45	1.65	1.85

CHAPTER IV

Cost Estimation

Introduction

We now describe the cost estimation exercise. The bulk of this chapter is concerned with the CBEAST program, which produces the cost data that are to be analyzed. We then briefly discuss the planning and execution of this exercise.

CBEAST generates data according to equation (2.7), which we re-write here for convenience:

$$(4.1) \qquad TC = FC + \left[\frac{a\ 100}{FPMID}\right]\left[\frac{Q\ FPI}{100}\right] + \left[\frac{b\ 10^{12}}{3\ FPMID}\right]\left[\frac{Q^3\ FPI}{10^{12}}\right] + ETC$$

$$= FC + a'\ W1 + b'\ W2 + ETC$$

The variable TC is total cost, and FC is fixed cost; both are in dollars. Units produced (and sold) is Q, and the index of the price of the variable factors of production is FPI. The variable factors are understood to be used in fixed proportions. The quantity ETC is a normally-distributed error term with user-supplied standard deviation. The various factors of ten are explained in Chapter II.

For small values of Q and large negative values of ETC, this cost function can produce observations of total cost which are less than FC. Since such values of total cost are difficult to explain, CBEAST ignores them and calculates another observation. In the event that more than NOB such values occur in the course of calculating NOB observations,

the trial is terminated and an error message printed. This message directs the user to reduce the standard deviation of RTC and/or to increase the minimum value of Q.

The principal output from this program is one or more sets of observations on total cost, TPI, price, and total revenue. The relation used to compute price and revenue from quantity is the demand curve that is to prevail during the play of the game. Its parameters are user supplied.

First the program calculates A and B in the following relation from the monopoly price when marginal cost is constant at $.50 ($P^M$):

$$(4.2) \qquad Q = A - B\,P + RMU$$

Here RMU is a normally-distributed error term with user-supplied standard deviation and mean zero. This equation is then used by the program to determine price, given Q and RMU. We shall discuss below what is done if the P so calculated is negative.

<u>Computer Input</u>

An input deck to CBEAST is made up as follows:

1. A <u>Title Card</u>, with any comment punched in columns 1 - 80. This comment will be printed at the top of each page of output.
2. One or more coefficient decks. Each of these consists of a <u>Coefficient Card</u> followed by one or more <u>Trial Cards</u>.

3. A Kill Card, with zeros punched in columns 1 - 2.

Table IV.1 shows the format of the Coefficient Cards. If KP is one, all data generated by the trials made with this set of coefficients will be punched just as printed, one observation to a card. If KP = 2, each variable printed will be punched separately. If KC is one, competitive profit is set equal to zero by adjusting the fixed cost. If KC is zero, fixed cost is equal to $50,000. This will resul in a competitive equilibrium with negative profits unless CMC = $.50.

If KS is one, the user must supply a starting point or seed for the random number generation routine on each trial card that follows this coefficient card. Usually, one will want to run a number of trials without punching data, to get a feel for the regression results different parameters produce. Once a good set of parameters has been located by this procedure, a deck will be punched and turned over to the students. The KS feature permits the user to duplicate the random sequence used in the first set of runs, since each trial's output contains the seed that was used tostart it. A seed may be any odd integer, preferably having four or more digits.

Monopoly price will typically be in the range $2.00 - $9.00; this is the range for which the game was designed. Capacity marginal cost should be greater than or equal to $.50.

Table IV.2 presents the format to be used in punching the Trial Cards. As we shall see immediately below, the larger are the autocorrelation coefficients, the smoother will be the FPI and Q series.

If it is desired not to use FPI, set FPMIN = FPMAX = 100; FPI will then not enter the expression for total cost. If this option is chosen, a value for ACCFP must still be punched, even though it is not a determinant of FPI observations in this case.

Data Generation

Successive observations on FPI and Q are generated according to the following formulae:

$$FPI(n) = (1 - ACCFP)(FPMIN + PRANGE\ PRAN) + ACCFP\ FPI(n - 1), \text{ and}$$

$$Q(n) = (1 - ACCQ)(QMIN + QRANGE\ QRAN) + ACCQ\ Q(n - 1), \text{ where}$$

$$PRANGE = FPMAX - FPMIN$$

$$QRANGE = QMAX - QMIN,$$

and PRAN and QRAN are random variables uniformly distributed between zero and one.

The generation of an observation for price is described by equation (4.2) above. The error term RMU is normally distributed with mean zero and standard deviation SDP = SDPIx1000. It is possible to choose a combination of maximum quantity (QMAX), monopoly price (PM), and standard deviation of RMU (SDPI) so that many of the prices calculated by this procedure are less than zero.. The larger are any of the parameters just mentioned, the more severe this problem becomes. Table IV.3 gives the largest value of SDPI for various values of PM and QMAX for which it is certain that fewer than 16% of the prices generated by CBEAST will be negative.

If a negative price is produced, CBEAST ignores it, obtains a new RMU, and calculates a new price. If NOB/2 negative prices are encountered in the course of generating NOB observations, the trial will be terminated and an error message printed.

Equation (4.1) is used to generate total cost from Q and FPI. The standard deviation of the error term RTC is equal to SDC = SDCIx1000.

Computer Output

All observations generated for total cost, total revenue, price, and factor price index are printed. In addition, all important input parameters and calculated parameters are shown. These include capacity marginal cost, monopoly price, the ranges and autocorrelation coefficients of the factor price index and of quantity, and the standard deviations of RMU and RTC. In addition, the seed for the random sequence is presented, in case the user wishes to reproduce the exact observations generated at some future date.

The next block of printed output gives the true values of FC, a', and b', as well as the values that would be produced by estimating equation (4.1) from the generated data. The usual statistics provided by a regression package are also presented. If CMC = $.50, b' = 0. In this case, the regression routine does not estimate b'.

The means of total cost, the factor price index, quantity, and price are shown. If KC was one for this trial, a message to that effect (and the calculated competitive profit) will be printed. All this information is given to enable the user to decide quickly whether the set of

data just generated is suitable for student use and to remind him of the inputs that went into that data. It should be a fairly simple matter to change the inputs to produce usable numbers.

One reminder: for use in decision making in the game, the R-squared is largely irrelevant. What matters is the quality of the coefficient estimates. If the estimated coefficients are significant and near their true values, the students will be able to use the estimated cost function successfully in the game.

If a deck is requested (if KP = 1 or 2 on the trial card), the generated observations will be punched as specified by KP. A title card stating "THIS CARD INDICATES THE BEGINNING OF A SET OF DATA" will be punched before the observations for each trial.

Subroutines Employed

In calculating regression output, CBEAST employs two generalized matrix manipulation subroutines written by Robert Hall. MULT performs matrix multiplication, and GMINV inverts square matrices.

RANDU is an IBM Scientific Subroutine Package program which generates quasi-random numbers distributed uniformly between zero and one. Input to the routine is an odd integer; output is the desired random number plus a second odd integer which is used as input the next time the subroutine is called. This feature makes it possible to reproduce the sequence of random numbers generated. This subroutine is specific to the IBM System/360; a new routine must be written if CBEAST is to be used on any other computer.

The subroutine RANGRC obtains uniformly distributed random numbers from RANDU and places two of them in Common storage to be used by the main program as PRAN and QRAN. RANGRC also places RMU and RTC in Common; these normally ditributed random numbers are calculated from four random terms obtained from RANDU. This routine is not specific to System/360.

The Exercise

The remarks made at the end of the last chapter about the demand estimation problem apply to the exercise discussed here. Handling the data can be important and should be carefully thought out. Grading should reward understanding of the relevant tools and their sensible application.

Most students seem to have a harder time with the cost problem than with the demand estimation. One difficulty is often the fact that an irrelevant variable, price, is supplied. There is a great temptation to use it in estimation. Alternatively, some students do not divide revenue by price in order to obtain quantity. Telling students at the start of the exercise that price is irrelevant to some extent defeats the purpose of the problem. A better approach is to confer with them about their estimates from time to time. They can be convinced rather easily that price is not relevant and that quantity is.

Another problem is the form of the cost function. Equations like (4.1) do not spring readily to mind. But the form used by CBEAST can be easily defended. First, the size of the plant is fixed, so the data is associated with a short-run cost function. This means that total cost

can be split into fixed and variable cost:

$$TC = FC + VC.$$

For given factor prices, VC depends only on Q. Thus if FPI is not allowed to vary, the best approach to estimating a cost function would involve a polynomial in Q.

The variable factors are assumed to be employed in fixed proportions. Thus if FPI doubles and Q is fixed, variable cost must double. (If labor is the only variable factor, FPI is a wage rate index.) So the cost function must be of the form

$$TC = FC + FPI\ G(Q)\ Q,$$

where G(Q) is variable factors used per unit of output. It would normally be expected to be a constant or an increasing function of Q. In CBEAST, $G(Q) = K + mQ^2$. Experimentation with alternative equations can logically involve only different forms of G(Q).

The best students will go through this "derivation" themselves. Most, however, will need to be led through the argument one way or another.

Table IV.1

Format of CBEAST Coefficient Cards

Columns	Variable	Description
1 - 2	NT	Number of trials to be made with this set of coefficients (right-justified, no decimal point).
3	KP	Punch a 1 if data generated with these coefficients is to be punched; punch a 0 if no punched output is requested from these trials.
4	KS	Punch a 1 if the following trial cards contain seeds for the random number routine. If 0 is punched, the program will supply starting terms in the random sequences.
5	KC	Punch a 1 if fixed cost is to be adjusted by the amount of competitive profit (or loss). Punch a 0 if this feature is not desired.
6 - 15	CMC	Capacity marginal cost = marginal cost when output is 100,000 units. (Punch a decimal point.)
16 - 25	PM	Monopoly price when marginal cost is constant and equal to $.50. (Punch a decimal point.)

Table IV.2

Format of CBEAST Trial Cards

Columns	Variable	Description
1 - 3	NOB	The number of observations (less than or equal to 200) to be generated in this trial. (right-justified, no decimal point).
4 - 11	FPMIN	Lower bound on the factor price index. (punch a decimal point)
12 - 19	FPMAX	Upper bound on the factor price index. (punch a decimal point).
20 - 27	ACCFP	Autocorrelation coefficient for the factor price index (punch a decimal point). Must be between zero and one.
28 - 35	QMIN	Lower bound on quantity (punch a decimal point.
36 - 43	QMAX	Upper bound on quantity (punch a decimal point).
44 - 51	ACCQ	Autocorrelation coefficient for quantity (punch a decimal point) Must be between zero and one.
52 - 59	SDCI	Standard deviation of the error term in the total cost equation, in thousands (punch a decimal point).
60 - 67	SDPI	Standard deviation of the error term used to compute price, in thousands. (punch a decimal point)
68 - 79	KU	Seed for random number routine. KU will not be read unless KS = 1 on the coefficient card. KU must be an odd integer; punch it right-justified without a decimal point.

Table IV.3

Maximum Recommended Values of SDPI in CBEAST

	QMAX				
PM	20,000	40,000	60,000	80,000	100,000
2.00	120.000	100.000	80.000	60.000	40.000
3.00	102.222	82.222	62.222	42.222	22.222
4.00	95.385	75.385	55.385	35.385	15.385
5.00	91.765	71.765	51.765	31.765	11.765

CHAPTER V

Forecasting and Game Initialization

This chapter is devoted to a description of the IGAME program. IGAME solves the problem of forecasting short-run and long-run competitive and monopoly equilibrium, using the true demand and cost functions. This program may also be employed to produce control cards for use in the play of the game. The next chapter presents the RGAME program, which uses these cards to simulate oligopolistic markets. We shall postpone our discussion of the game exercise until then.

We shall not devote a separate section to the forecasting problem, simply because there is not much to say. Students have a variety of difficulties; many result from a confusion about which demand curves to use. Short papers should be required, and method is clearly more important than results. It has been my practice to have students confine arithmetic to appendices, which I examine only if their answers are quite different from the IGAME results. I think the forecasting exercise is very valuable but it seems to be the most difficult for students.

Program Input

An input deck to IGAME is made up as follows:

1. One IGAME Title Card (See Table IV.1)
2. NI IGAME Industry Cards (See Table V.2)

The quantity NI is defined on the Title Card, as shown in Table V.1. If more or fewer than NI industry cards are present, the program will not execute properly.

Most of the variables on the Title Card are self-explanatory. The quantity KP is present because it may be desired to use IGAME only to provide the solution to the forecasting problem, rather than to initialize a game exercise. When IGAME is used to initialize the game, the period number NP, will almost always be one.

The industry cards are a bit more complicated. The quantities NST and MO determine the sort of output that will be received by student firms operating in the industry in question. Table V.3 provides a complete listing of theoutput codes; the instructor has a great deal of control over what information is available to the firms.

Firms operating in some industries may be permitted to make and receive lump-sum cash transfers. Payments made to a firm ineligible to receive them will be debited to the payer but not credited to the payee. Payments made by a firm ineligible to make them will also be debited. Negative side payments will be treated as positive. The variables KSP determines the legality of side payments for each industry.

At the discretion of the instructor, it may be possible for outside firms to move into some industries. A lump-sum payment is assessed each time a firm switches industries; the amount depends on the industry entered. Any attempt to enter an industry into which entry is not allowed will result in the firm being restored to the industry in which it last operated.

Any attempt to enter an industry into which entry is not allowed will result in the firm being restored to the industry in which it last operated. Any attempt to enter an industry with ten firms already present will be similarly frustrated, as PGAME is unable to handle indu stries with more than ten firms. The variable KEC determines whether or not entry will be allowed. If entry is permitted, the amount of the lump-sum entry payment will be calculated from either ECA or ECP.

For each industry, the variables KC and CMC are used to determine the cost structure. Similarly, the demand curve is determined by PM, F and either PC or FK. (We shall discuss the algebra involved below.)

Of the last four quantities on the industry cards, SDD and SDC are self-explanatory. The initial entry factor, F, enters into the determination of the demand structure as shown below. It is used to raise or lower firms' demand curves. Finally, NP is the number of plants that the students are to assume in the industry for the forecasting exercise. It is used by the program only in the computation of long-run equilibria.

Computation

Each firm's total cost is given by an equation of the form of (2.6) which we shall re-write in the notation of IGAME as

(5.1) $$TC = FCI + AMC\ Q + (BMC/3)\ Q^3 + u,$$

where u is an error term with standard deviation given by SDC. IGAME first calculates demand and marginal cost parameters for each industry. AMC is always equal to .50, and BMC is given by

$$(5.2) \qquad BMC = (CMC - .50/)10^{10}$$

Fixed cost, FCI, cannot be determined until the short-run competitive equilibrium price and quantity have been computed.

Each firm faces a demand function of the form of (2.3). Re-writing this in the notation used by IGAME, we have

$$(5.3) \qquad Q = AF - BF\ P + CF\ \overline{PO} + e,$$

where the standard deviation of the error term e is determined by SDD. We must now consider how AF, BF and CF are determined by PM and PC or FK.

Critical price, PC, and FK are two equivalent ways of expressing the responsiveness of firms' demand to differences in P and $\overline{PO}$. Both can be easily interpreted when marginal cost is \$.50. In this case, let P = $\overline{PO}$-X, and V equal the derivative of total profits with respect to X, ignoring the capacity constraint, evaluated at X equals zero. Then PC is the value of $\overline{PO}$ for which V is zero; it is the price at which the firm has nothing to gain from under-cutting the competition, even without capacity restrictions. V will be positive for all $\overline{PO}$ greater than PC. The quantity FK is the value of V at $\overline{PO}$ = 1.00.

If FK is given, the program proceeds directly to compute the demand parameters. If PC is supplied instead, IGAME computes FK according to

$$FK = \frac{400,000\ (2PM - 1)\ (1 - PC)}{(4PM - 3)\ (2PC - 1)}$$

Given FK and PM, the next step is to calculate

$$A = \frac{100,000\ (4PM - 1)}{(4PM - 3)}$$

$$(5.4) \qquad B = 2\ FK + 200,000$$

$$C = 2\ FK + \frac{800,000\ (PM - 1)}{(4PM - 3)}$$

Normally these coefficients would simply be used as AF, BF and CF in equation (5.3). If this is done, it will be the case that if all firms charge \$1.00 , all will receive, on average, demands of 100,000 units.

The instructor may wish to raise or lower the demand curve, so that this condition does not hold. The initial entry factor, F, is used to do this. The actual demand parameters used in RGAME are obtained from equations (5.4) and

$$AF = A\ F$$

$$(5.5) \qquad BF = B\ F$$

$$CF = C\ F$$

If F = .50 and the number of firms initially present is four, for example, they will share the total demand that would normally accrue to two firms. It is as if there had been entry prior to period one. Note that the standard deviation of the demand curve error term is also multiplied by F.

After computing these coefficients, IGAME obtains the solutions to the forecasting problem. The short-run competitive and monopoly solutions are obtained by setting short-run marginal cost equal to price and marginal revenue, respectively. Note that in both situations, $P = \overline{PO}$, so only AF and (CF-BF)are used to determine demand. Note also that NP is not used here.

If the short-run competitive solution is required to have zero profit, the fixed cost is adjusted accordingly, and a message is printed. If KC = 0, fixed cost remains at $50,000. In the monopoly case, the standard deviation of profit is calculated and shown; it depends on the standard deviations of the cost and demand errors in a straight-forward fashion.

The next step is to compute the long-run competitive and monopoly equilibria. Here NP is used, as the equilibrium number of plants must be integer. In long-run competitive equilibrium, price is (approximately) equal to minimum average total cost and profits are non-negative. In long-run monopoly equilibrium, marginal

revenue is (approximately) equal to minimum average total cost. In the monopoly case, the most profitable integer number of plants is found. Under competition, the equilibrium number of plants is chosen so that the short-run equilibrium involves non-negative profits, but the entry of one more firm would drop profits below zero.

Two more sets of figures are produced. First, consumer's surplus is used to evaluate the short-run and long-run welfare loss due to monopoly. Second, the short-run Edgeworth-Bertrand-Quandt equilibrium is computed. This is a Cournot-like non-collusive equilibrium in which each firm assumes that $\overline{PO}$, the average of its competitors prices, will remain unchanged. Each firm then chooses P to maximize its own profits. Equilibrium obtains when $P = \overline{PO}$. In the game, one might expect such equilibrium position to be reached in the absence of collusion. This computation employs the values of BF and CF in (5.3), not just their difference. Thus the Bertrand equilibrium depends, as the competitive and monopoly equilibrium do not, on FK or PC. The larger is FK or the smaller is PC, the lower will be the Bertrand equilibrium price.

Program Output

For each industry, IGAME produces a page of printed output with three main sections. The first block repeats the input parameters from the industry and title cards. The next section gives cost and demand parameters that will be used in BGAME. The final block contains the market equilibrium solutions and the welfare loss due to monopoly.

If KP on the IGAME title card was equal to one, IGAME will punch

control cards for use in RGAME. It will first punch two RGAME Industry Cards for each industry, then a card with "999999" in the first six colums (a Delimiter Card), and then an RGAME Title Card. Tables VI.1 and VI. 2 describe these cards.

Computer Details

IGAME is composed of a main program and four subroutines. Subroutine CALQ computes competitive or monopoly output. A code variable L, is used to tell CALQ which case to consider. Subroutines CALME and CALCE calculate long-run monpoly and competitive equilibrium, while CALBE computes the Bertrand equilibrium.

Table V.1

IGAME Title Card Format*

Columns	Variables	Explanation
1	KP	Punch a "1" if a deck of control cards for RGAME is to be punch, zero otherwise.
2 - 3	NCT	Number of copies of IGAME printed output.
4 - 5	NP	Period number for RGAME control cards - usually "1".
6 - 7	NI	Number of industries present.
8 - 9	NCI	Number of copies of instructor's output to be printed by RGAME.
10 - 20	KU	Seed for random number generation in RGAME. Punch "0" in column 20 if the default value of the seed is to be used. Otherwise KU must be an odd integer greater than 1,000.
21 - 80	TITLE	Any comment - will be printed at the top of each page of IGAME (and RGAME) output.

* Punch by the instructor as input to IGAME. All Numbers on this card should be punch right-justified without decimal points.

IGAME Industry Card Format*

Columns	Variable	Explanation
1 - 2	IN	Industry number
3 - 4	NFS	Number of firms to be initially present in the game.
5 - 6	NST	Number of copies of student output to be produced by RGAME.
7 - 8	KO	Output code 1.
9 - 10	KO	Output code 2.
11 - 12	KO	Output code 3.
13	KC	Punch "1" if fixed cost is to be set so as to yield zero profits at competitive equilibrium If fixed cost is to be left at $50,000, punch "0".
14	KSP	Punch "1" if firms in this industry are to be allowed to make and receive side payments involving other firms. If "0" is punched, side payments will be illegal in this industry in RGAME.
15	KEC	If "0" is punched, entry into this will be blocked. If "2" is punched, entry cost will be calculated from ECP and if "1" is punched, entry cost will be calculated from ECA.
16 - 21	CMC	Marginal cost when output is equal to capacity (=100 ,000 units).
22 - 27	PM	Monopoly price when marginal cost is equal to $.50.
28 - 33	PC	Critical price (read only when FK = 0).

Table V. 2 continued.

Columns	Variable	Explanation
34 - 45	PK	(Calculated from PC if zero).
46 - 52	ECA	Absolute entry cost in thousands of dollars.
53 - 59	ECP	Entry cost as a percentage of monopoly profit.
60 - 66	F	Initial entry factor
67 - 71	SDD	Standard deviation of error term in the demand equation. In thousands.
72 - 76	SDC	Standard deviation of the error term in the cost equation. In thousands
77 - 80	NP	Number of plants originally present in the forecasting problem.

*Punched by the instructor as input to IGAME.
The numbrs in columns 1 - 15 and 77 - 80 must be right-justified without decimal points; all numbers in columns 16 - 76 must contain decimal points.

Table V.3

Output Code 1: Other Industries

K01		K02	
4	Number of firms average price	4	Total sales and total profits.
3	Average price only	3	Total profits only
2	Number of firms only	2	Total sales only
1	None of this information	1	None of this information

Output Code 2: Own Industry

K03		K04	
4	Number of firms, average price	4	Total sales and total profits
3	Average price only	3	Total profits only
2	Number of firms only	2	Total sales only
1	None of this information	1	None of this information

Output Code 3: Competitors

K05

5 Identify all firms, given all prices.

4 Identify only entrants, give all prices

3 Identify all firms, give no prices

2 Identify only entrants, give no prices

1 None of the above information.

Table V.3 (continued).

Output Code 3: Competitors

K06

8 Sales, unfilled orders, profits

7 Sales, profits

6 Sales, unfilled orders

5 Unfilled orders, profits

4 Sales

3 Unfilled orders

2 Profits

1 None of the above information

*These six digits must be punched in columns 7 - 12 of the IGAME industry cards; they will serve as input to RGAME.

Chapter VI

Play of the Game

The main purpose of this chapter is to describe RGAME, the computer program that operates the game exercise. This is a rather complicated program. - not because it performs complex computations, but because it keeps track of a great deal of information. It uses the output from IGAME to define industries' structures, and it uses students' inputs to generate outputs and profits for the various firms. This program can handle a maximum of ten industries, with a maximum of ten firms operating in each.

Besides the student firms, there is a dummy firm present in the game. Its number if 9999. This firm acts as the instructor's fiscal agent, and it can make side payments and levy fines. In addition, it can receive payments from any firm.

The chapter concludes with a discussion of the game exercise. Some thoughts on setup, operation, and evaluation are presented.

Program Input

An input deck to RGAME is made up as follows:

1. One RGAME Title Card (Table VI.1).
2. RGAME Industry Cards (Table VI.2; two for each industry).
3. One Delimiter Card ("99" in columns 1 - 2). (The Delimiter Card punched by IGAME may be used.)

4. RGAME Average History Cards (Table VI.3; omit in period one).

5. One Delimiter Card ("99" in columns 1 - 2; one in period one).

6. RGAME Firm Cards (Table VI. 4; one for each firm).

7. One Dummy Firm Delimiter Card (Table VI. 5).

8. Dummy Firm Side Payment Cards (Table VI.5; must be as many as specified on the Dummy Firm Delimiter Card).

9. One Delimiter Card ("99" in columns 1 - 2). (omitif no dummy firm side payments).

The deck must be made up exactly as shown here. The industry cards may be in any order, but the wrong number of industry cards will cause RGAME to terminate with an error message. Similarly, the order of the Average History cards does not matter, but there must be one for each firm and one for each industry. There must be one firm card for each firm. It is advisable to shuffle these before each run, as students in industries in which little or no information is printed may deduce the identity of their competitors if the order of firm cards is always the same. The number of Dummy firm Side Payment Cards must correspond to the number specified on the immediately preceeding Delimiter Card.

Both the Industry Cards and the first period's Title Card are punched by IGAME. Notice though that the Title Card is punched last by IGAME- it must be moved to the front of the deck before IGAME output can be used in RGAME. RGAME punched Average History cards during each run to be used as input for the next run, as well as producing a new Title Card. There are no Average History Cards for period one, so items 4 and 5

in the input list above must be omitted for the first period. The Firm Cards are submitted by the students before each run.

The instructor's main responsibility is to handle the cards punched by the computer and by the students and to prepare items 8 and 9 in the list above. It is through these cards that he, acting as dummy firm number 9999, can make payments to or levy fines upon any of the firms in the game.

Computation

When the input deck is being read, checks for the number of cards present in items 2, 4, 6, and 8 are preformed. The next task is to place firms in industries, charge and credit entry costs and side payments, and to administer dummy firm side payments. Part of the instructor's output is a side payment summary. A comment colum in this table indicates if a given side payment was an entry cost (paid by the entrant to firm number 9999), if it was illegal, or if it was made to a non-existent firm. Also a firm which has made such an invalid side payment or attempted entry into an industry which is blocked, non-existent, or filled (ten firms already present) will receive an error message on its student output sheet.

Side payments which are illegal or are designated for non-existent firms are debited to the payer but not credited to the payee. If the instructor does not desire this feature, he can make an appropriate side payment from the dummy firm in the following period.

Once all firms have been placed in the appropriate industries, demand parameters are adjusted for entry and exit (see equation (2.4), and all fixed costs are adjusted to include side payments and entry costs. RGAME then generates quasi-random numbers to serve as cost and demand error terms and it computes sales, unfilled orders, and profits for each firm.

The adjusted demand equations are used to compute the demand for the output of each firm. Firms for which demand is greater than 100,000 units sell 100,000 units and have unfilled orders equal to the difference between their demand and 100,000. These firms are then put to the side and simulation program divides their unfilled orders among those firms with excess capacity and positive demand. To do this, it first calculate the total demand in the industry. It then computes the share each of the firms with positive sales and unfilled orders received of this total demand. Each then receives a fraction of total unfilled orders equal to its share of total industry demand. Thus if any firm's demand is zero, it will receive no unfilled orders. If the firms with excess capacity accounted for only half of industry demand, they will receive only half of the unfilled orders generated by the other firms. If this allocation raises some demands above 100,000 units, the unfilled orders so produced are allocated in this same fashion.

The basic idea underlying this simple model is that a dissatisfied retailer does not immediately withdraw his orders. He places some additional orders with companies having excess capacity (and chargin higher prices), but his demand is reduced. The process continues untill all orders are met or untill all firms, have unfilled orders. If there are no unfilled orders generated, total inudstry sales will obviously equal total industry demand. If any firms have unfilled orders, though, total industry sales will be less than total industry demand .

For the instructor's convenience, firms and industries are rated according to their profits. This is done because it may otherwise be difficult to compare industries where monopoly profits differ. Let S* be the total positive side payments received by the firm or industry in question from firm 9999. Then the A ratings for firm II JJ and inudstry II are defined as follows:

$$\text{Industry} = \frac{100\ (\text{actual profits earned in II} - S^*)}{MP^*\ (II)\ NO(II)}$$

$$\text{Firm} = \frac{100\ (\text{profit of II JJ} - S^*)}{MP^*\ (II)}$$

The quantity MP* (II) is the monopoly profit that could have been earned in industry II, assuming no entry or exit, and NO (II) is the number of firms originally present inII. The A rating for a firm thus compares its performance, no matter what industry it is operating in, with the profits

it could have earned had it been a member of a well-run cartel in its original industry. Similarly, the industry A rating ignores the impact of entry in the numerator, though entry will have an effect on the denominator.

The B ratings for industry II and firm II JJ operating in industry KK are defined as follows:

$$\text{Industry} = \frac{100\ (\text{actual profits earned in II} - S^* + EC)}{MP^{**}(II)\ NA(II)}$$

$$\text{Firm} = \frac{100(\text{profit of II JJ} - S^* + EC)}{MP^{**}(KK)}$$

Here MP**(II) is the profit that could be earned by a member of a well-run cartel composed fo the firms actually present in industry II: MP**(II) reflects entry and exit if they have occurred. The quantity NA(II) is the number of firms currently operating in industry II, and EC is the amount of entry cost payments made by the firm or industry in question.

Several features of these measures should be mentioned. First, the A rating is primarily a long-run measure, while the B ratings reflect mainly short-run performance. The A ratings compare profits with an absolute standard, while the B ratings take as given the number of firms in and moving into each industry. Second, neither rating takes into account the error terms experienced by particular firms. The ratings will

on average, reflect performance moderately well as long as the standard deviations of the error terms are no too large. Finally, the treatment of S* may seem a bit obscure. Positive side payments are distinguished from negative side payments made by the dummy firm so that the instructor can determine whether dummy firm payments should enter the ratings or not. If he wishes them ignored, he can make only positive side payments from firm 9999. If, on the other hand, he wants his side payments to matter in the ratings, he can have 9999 make only negative side payments (levy fines and not provide subsidies). Since it is really only relative profits that matter, the instructor can bring about any desired result by using only positive or only negative side payments.

Program Output

Each run of RGAME produces one RGAME Title Card and a set of RGAME Average History Cards, one for each firm and one for each industry, to be used as input to the next RGAME run. In addition, a set of Period History Cards are punched, one for each firm and one for each industry. These cards are described in Table VII.2. They are used as input to SGAME. The three blocks of cards are punched in the order mentioned. They are separated by distinctively punched cards.

Each set of instructor's output consists of a side payment summary, described above, a firm directory, and one sheet for each industry. The industry in which firm II JJ is currently operating is the number appearing the row II, column JJ of the firm directory matrix.

Each industry sheet in the instructor's output indicates which firms were present in the industry, and where they operated in the previous period. Each firm's price, sales, profit, unfilled orders, rating A, and rating B are presented, along with the industry average price, total sales, total profit, total unfilled orders, rating A, and rating B. The averages to date of all these quantities are also shown.

Students' output may contain either a great deal of information or almost none at all; see Table V.3 for details. At a minimum, all firms are shown their own price, sales, profit, and unfilled orders. No time averages are present on students' output.

Computer Details

RGAME consists of a main program plus five subroutines. The main program calls the various subroutines, updates averages, and punches Average History and Period History Cards. Subroutine READP handles all input to the program. It checks for input errors as described above, prepares data in common storage, places firms in their current industries, charges and credits entry costs and side payments (checking for existence and legality as described above), and administers dummy firm side payments. All lump-sum payments are added to or subtracted from fixed cost.

RANDU is an IBM Scientific Subroutine Package program which generates quasi-random numbers distributed uniformly between zero and

one. Input to the routine is an odd integer; output is the desired random number plus a second odd integer which is used as input the next time the subroutine is called. The odd integer produced by the last call to RANDU in each PGAME run is punched on the title card as input to the next period's run. This routine will work properly only on IBM System/360 computers.

The main program procures uniformly distributed random numbers from RANDU, converts them to normally distributed random numbers, and uses them to adjust the intercept of each firm's demand function and its fixed cost.

The key routine CALC finds sales, profits, and unfilled orders. It is here that demands are calculated, unfilled orders allocated, and p rofits generated. Subroutine RATE computes A and B ratings for firms and inustries, as described above.

Finally, PRNTO handles all printed output from PGAME, including error messages. If multiple copies of students' or instructor's output are requested, they will be produced in blocks rather than one page at a time. PRNTO deciphers theoutput codes while printing students' output.

The Exercise

First, we shall repeat a point made in Chapter II. The first two digits of the firm number refer to the industry the firm began in, and the second two are the firm's number within the industry. If there are to be NN industries they must be numbered 01, 02, ..., NN. If industry JJ begins with II firms, they must be numbered JJ01, JJ02, ..., JJII. Exo-

erience suggests that the fewer students in each firm, the more they will learn from the exercise. It also seems that the more frequently decisions must be submitted, the greater the learning. Daily play with one-man firms is optimal.

When industries are being set up, this should be done so as to provide structural contrasts. This way students and instructor can examine the impact of structure on performance. IGAME permits differences in number of firms, sensitivity of demand to intra-industry price differences, entry conditions, side payment opportunities, and legal structures (discussed in the Student's Manual). Also, as mentioned in Chapter II, cost and industry demand can differ between industries. To be sure of the "right" results, make the contrasts fairly strong.

One final remark on setup. The standard deviations of the costs and demand error terms should be kept small - generally smaller than in the estimation exercise. This ensures more sensible comparisons of firms' performances, and it will enable the students to analyze their experience in the game as it progresses with more confidence. Remember that they do not know and must estimate the effects of intra-industry price differences.

The main problem students seem to have regarding game output is the relations between demand, sales and unfilled orders. The

Student's Manual attempts to clarify this point, but there will still be questions. Also, students often punch their Firm Cards incorrectly. These must be checked before running PGAME to see if all numbers are in the correct columns. One remedy for carelessness is to levy fines for mispunched cards.

It is very important to maintain tight security when cards are handed in. Students become very competitive during the game, and if possibilities for sabotage of other firms exist they will be exploited. One approach is to require identification from students submitted cards, but this can be time-consuming. A better method is to obtain a cabinet or box that can be locked and which has a slit that permits the insertion of punched cards. Issue each firm a password that only they know. Require that all cards have handwritten on them the correct password, the date, and the time of day. Remember that once a card has been submitted it remains in force until another is turned in; firms need not submit a card for each period of play.

Grading should be based partly on profits earned, to provide an incentive to rational play. Also, students should be required to write short papers on the exercise. They should be discouraged from a "This is what my firm did and why" approach. Rather, they should discuss the determinants of the relative performances of the various industries. Many of these will relate to the structural elements you

built into the game, but personalities and attitudes towards risk also enter. Before having students write papers, they should be given the output from a run of SGAME, discussed in the next chapter.

Table VI.1

RGAME Title Card Format*

Columns	Variables	Explanation
1 - 2	NP	Period Number
3 - 4	NI	Number of industries
5 - 6	NCE	Number of copies of instructor's output.
7 - 9	NFT	Total number of firms
10 - 20	KU	Seed for random number generation
21 - 80	TITLE	Any comment - appears at the top of each page of RGAME output.

*The title card for period one is punched by IGAME, along with the industry cards. Title cards for subsequent periods are generated by RGAME. None of the numbers on this card have decimal points.

Table VI.2

RGAME Industry Card Format*

First Card

Columns	Variable	Explanation
1 - 2	IN	Industry number.
3 - 4	"00"	Code indicating first card
5 - 6	NFS	Number of firms initially present
7 - 8	NST	Number of copies of student output
10	KSP	One if side payments are legal, zero otherwise.
11 - 22	AF	Demand parameters; see equation (1)
23 - 34	BF	
35 - 46	CF	
47 - 58	BMC	Cost parameters; see equation (2)
59 - 66	AMC	

Second Card

Columns	Variable	Explanation
1 - 2	IN	Industry number
3 - 4	"01"	Code indicating second card
5 - 10	KO	Output Codes; see Table 3
11 - 22	PIMS	Monopoly profit
23 - 34	ECA	Entry cost.
35 - 46	SDDF	Standard deviation of demand error term
47 - 58	SDC	Standard deviation of cost error term.
59 - 66	FCI	Fixed cost

Table VI.2 (continued)

* These cards are punched by IGAME and used in all RGAME and SGAME runs. All fields starting with column 11 have decimal points.

Table VI.3

RGAME Average History Card Format*

Columns	Variable	Explanation
1 - 2	NPL	The period during which the card was punched; this is when the averages stopped.
3 - 4	II	Industry number
5 - 6	IF	If this is a firm card, IF is the last two digits of the firm name. IF is zero to indicate industry Average History cards.
7 - 8	IL	For industry cards, IL = 00. For firm cards, IL is the number of industry the firm was operating in during period NPL.
9 - 20	AVIP or AVP	Average price
21 - 32	AVSI or AVS	Average sales
33 - 44	AVUOI or AVUO	Average unfilled orders.
45 - 56	AVPII or AVPI	Average profit
57 - 68	AVRAI or AVRA	Average A rating
69 - 80	AVRBI or AVRB	Average B rating

*These cards are punched by each run of RGAME and used as input into the next RGAME run. They are not needed as input into the run labelled period one.

Table VI.4

RGAME Firm Card Format*

Columns	Variable	Explanation
1 - 2	II	The four-digit firm name, no decimal.
3 - 4	IF	
5 - 6	IC	Industry the firm wishes to operate in this period May be left blank if no change in industry is desired.
7 - 12	AP	Current price. Punch a decimal point.
13 - 14	JI	The four-digit name of the firm to which a side payment is to be made. May be left blank if no side payment is to be made.
15 - 16	JJ	
17 - 26	S	Amount of the side payment in dollars. Punch a decimal point. This may also be left blank. (If negative, will be made positive.)

*These cards are turned in by students for each period RGAME is run.

Table VI.5

RGAME Dummy Firm Card Format*

Delimiter Cards

Columns	Variable	Explanation
1 - 2	"99"	This indicates the end of the firm card section of input and the beginning of the dummy firm card section.
15 - 16	NDSP	The number of side payment cards to follow. Punch right-justified, no decimal.

Side Payment Cards

Columns	Variable	Explanation
1 - 2	I	If I and J are both non-zero, the payment is made to firm number IJ. If only J is zero, the payment is made to all firms <u>originating in</u> industry I- that is, to all firms numbered IK, for any K. If only I is zero, the payment is made to all firms <u>operating in</u> industry number J.
3 - 4	J	
5 - 16	SDP	Amount of the side payment, in dollars. May be negative. Punch a decimal point.

*These cards are punched by the instructor as input to RGAME.

CHAPTER VII

Summarizing the Game

This chapter is a brief discussion of SGAME, the program used to translate the RGAME Period History cards into a summary of the results of a game exercise. This program should be run at the end of any game exercise, and the output should be given to students before their papers on the exercise are due.

SGAME consists of a single main program. No subtractives are called. The program can handle a maximum of 30 periods of RGAME output.

Program Input

An input deck to SGAME is made up as follows:

1. One SGAME Title Card (Table VII.1)
2. RGAME Industry Cards (Table VII.2; two for each industry).
3. One Delimiter Card ("99" in columns 1 - 2).
4. SGAME Period History Cards (Table VII.2; for each period of play covered there must be one card for each firm and one for each industry).
5. One Delimiter Card ("99" in columns 1 - 2).

The deck must be made up exactly as shown here. The Industry and Period History cards may be in any order, as long as they are in the proper location in the input deck. The wrong number of either type of card will cause execution to be terminated and an appropriate error message printed.

The instructor punches only the Title Card and the Delimiters. Industry Cards are punched by IGAME and Period History Cards are punched by RGAME. All the quantities on the SGAME Title Card are self-explanatory.

SGAME does very little computation. Its main tasks are to sort out the information on the Period History Cards by firm and period and by industry and period. It then calculates averages of most quantities over the number of periods of play.

Program Output

As the Title Card indicates, there are four types of printed output. If it is desired to suppress printing of one or more of them, the appropriate variable or variables on the Title Card should be set to zero. If multiple copies of any output type are desired, they will be produced as a block rather than one sheet at a time.

Instructor's industry sheets give the number of firms present, industry average price, total sales, total profit, total unfilled orders, rating A, and rating B for each period. Averages over time of all these quantities are printed. The number of firms initially present and the legality of side payments and entry are also shown. If entry was permitted, the entry cost is printed. Students' industry sheets contain all of this information except for the ratings.

The instructor's firm sheets trace the history of each firm from period to period. The industry it operated in each period is printed, along with the number of firms that began in that industry and the number of firms currently present. The firm's price, the industry

average price, the firm's sales, unfilled orders, profits, rating A, and rating B are shown for each period. Averages over time of all quantities are presented. Again, the students' firm sheets contain all of the information on the instructor's sheets except for the ratings.

Table VII.1

SGAME Title Card Format

Columns	Variable	Explanation
1 - 2	NPT	Total number of periods played.
3 - 4	NI	Number of industries
5 - 7	NFT	Total number of firms
8 - 9	NII	Number of sets of instructor's industry sheets desired
10 - 11	NIF	Number of sets of instructor's firm sheets desired.
12 - 13	NSI	Number of sets of students' industry sheets desired.
14 - 15	NSF	Number of sets of students' firm sheets desired
17 - 80	TITLE	Any comment - will appear at the top of each page of output.

*This card is punched by the instructor as input to SGAME . All numbers must be right-justified with no decimal points.

Table VII.2

SGAME Period History Card Format

Columns	Variable	Explanation
1 - 2	NP	The period number during which the card was punched.
3 - 4	II	Industry number
5 - 6	IF	If this is a firm card, IF is the last two digits of the firm name. IF is zero to indicate industry history cards.
7 - 8	I	For industry cards, I is the number of firms present during period NP. For firm cards, I is the industry in which firm II IF operated in period NP.
9 - 20	AP or P	Industry average price or firm price.
21 - 32	TS or ASA	Industry total sales or firm sales.
33 - 44	UOI or UO	Industry total unfilled orders or firm unfilled orders.
45 - 56	TRP or PI	Industry total profits or firm profits.
57 - 68	RAI or RAF	Industry or firm rating A.
69 - 80	RBI or RBF	Industry or firm rating B.

*These cards are punched by RGAME as input into SGAME.

Date Due